PCT/JP98/02435
09/445088
21.07.98

日本国特許庁
PATENT OFFICE
JAPANESE GOVERNMENT

REC'D 31 JUL 1998
WIPO PCT

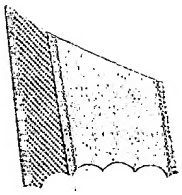
別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: １９９７年　６月　３日

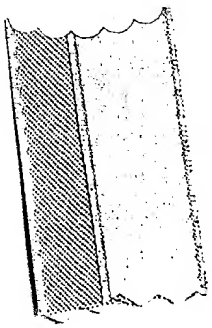
出願番号
Application Number: 平成　９年特許願第１４４９１６号

出願人
Applicant (s): 株式会社日立製作所



PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)



１９９８年　６月　５日

特許庁長官
Commissioner,
Patent Office

荒井寿光

出証番号　出証特平１０－３０４３２０５

【書類名】　特許願

【整理番号】　H97017371A

【提出日】　平成 9年 6月 3日

【あて先】　特許庁長官　殿

【国際特許分類】　H04N　7/13

【発明の名称】　フレーム間予測画像の合成方法及び該合成方法を用いた画像符号化方法、画像復号化方法

【請求項の数】　35

【発明者】

【住所又は居所】　東京都国分寺市東恋ケ窪一丁目２８０番地
株式会社日立製作所中央研究所内

【氏名】　中屋　雄一郎

【特許出願人】

【識別番号】　000005108

【氏名又は名称】　株式会社日立製作所

【代理人】

【識別番号】　100068504

【弁理士】

【氏名又は名称】　小川　勝男

【電話番号】　03-3212-1111

【手数料の表示】

【予納台帳番号】　013088

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9003094

【プルーフの要否】　要

【書類名】明細書

【発明の名称】フレーム間予測画像の合成方法及び該合成方法を用いた画像符号化方法、画像復号化方法

【特許請求の範囲】

【請求項１】

画素のサンプリング間隔を水平，垂直方向共に１として、サンプリング点が座標の水平，垂直成分が、共に整数にｗを加えた数である点の上に存在している画像を対象として（ただし、ｗ＝ｗｎ／ｗｄ、かつｗｎは負ではない整数、かつｗｄは２のｈｗ乗、かつｈｗは負ではない整数、かつｗｎ＜ｗｄ）、

４個の代表点における動きベクトルに対し、共１次内・外挿を行うことによって画素の動きベクトルを計算する場合に、

座標（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）（ｉ，ｊ，ｐ，ｑは整数）に代表点が存在し、

かつ代表点の動きベクトルの水平・垂直成分が１／ｋの整数倍の値をとり（ただし、ｋは2のｈｋ乗、かつｈｋは負ではない整数）、

かつ座標（ｘ＋ｗ，ｙ＋ｗ）に位置する画素の動きベクトルを求めるときに、座標（ｉ，ｊ）と（ｉ，ｊ＋ｑ）に位置する代表点の動きベクトルに対して線形内・外挿を行うことにより、座標（ｉ，ｙ＋ｗ）に位置する点の動きベクトルの水平・垂直成分をそれぞれ１／ｚの整数倍をとる数値として（ただし、ｚは2のｈｚ乗、かつｈｚは負ではない整数）求め、さらに座標（ｉ＋ｐ，ｊ）と（ｉ＋ｐ，ｊ＋ｑ）に位置する代表点の動きベクトルに対して線形内・外挿を行うことにより、座標（ｉ＋ｐ，ｙ＋ｗ）に位置する点の動きベクトルの水平・垂直成分をそれぞれ１／ｚの整数倍をとる数値として求めた後に、（ｉ，ｙ＋ｗ）と（ｉ＋ｐ，ｙ＋ｗ）に位置する上記２個の動きベクトルに対して線形内・外挿を行うことにより、座標（ｘ＋ｗ，ｙ＋ｗ）に位置する画素の動きベクトルの水平・垂直成分をそれぞれ１／ｍの整数倍をとる数値として（ただし、ｍは2のｈｍ乗、かつｈｍは負ではない整数）求めることを特徴とするフレーム間予測画像の合成方法。

【請求項２】

画素のサンプリング間隔を水平，垂直方向共に１として、サンプリング点が座

標の水平，垂直成分が、共に整数にｗを加えた数である点の上に存在している画像を対象として（ただし、ｗ＝ｗｎ／ｗｄ、かつｗｎは負ではない整数、かつｗｄは２のｈｗ乗、かつｈｗは負ではない整数、かつｗｎ＜ｗｄ）、
４個の代表点における動きベクトルに対し、共１次内・外挿を行うことによって画素の動きベクトルを計算する場合に、
座標（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）（ｉ，ｊ，ｐ，ｑは整数）に代表点が存在し、
かつ代表点の動きベクトルの水平・垂直成分が１／ｋの整数倍の値をとり（ただし、ｋは2のｈｋ乗、かつｈｋは負ではない整数）、
かつ座標（ｘ＋ｗ，ｙ＋ｗ）に位置する画素の動きベクトルを求めるときに、座標（ｉ，ｊ）と（ｉ＋ｐ，ｊ）に位置する代表点の動きベクトルに対して線形内・外挿を行うことにより、座標（ｘ＋ｗ，ｊ）に位置する点の動きベクトルの水平・垂直成分をそれぞれ１／ｚの整数倍をとる数値として（ただし、ｚは2のｈｚ乗、かつｈｚは負ではない整数）求め、さらに座標（ｉ，ｊ＋ｑ）と（ｉ＋ｐ，ｊ＋ｑ）に位置する代表点の動きベクトルに対して線形内・外挿を行うことにより、座標（ｘ＋ｗ，ｊ＋ｑ）に位置する点の動きベクトルの水平・垂直成分をそれぞれ１／ｚの整数倍をとる数値として求めた後に、（ｘ＋ｗ，ｊ）と（ｘ＋ｗ，ｊ＋ｐ）に位置する上記２個の動きベクトルに対して線形内・外挿を行うことにより、座標（ｘ＋ｗ，ｙ＋ｗ）に位置する画素の動きベクトルの水平・垂直成分をそれぞれ１／ｍの整数倍をとる数値として（ただし、ｍは2のｈｍ乗、かつｈｍは負ではない整数）求めることを特徴とするフレーム間予測画像の合成方法。

【請求項３】

座標（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）に位置する代表点の動きベクトルの水平・垂直成分をｋ倍したものである（ｕ0，ｖ0）、（ｕ1，ｖ1）、（ｕ2，ｖ2）、（ｕ3，ｖ3）を用いて座標（ｘ＋ｗ，ｙ＋ｗ）に位置する画素の動きベクトルを求めるときに、
座標（ｉ，ｙ＋ｗ）に位置する点の動きベクトルの水平・垂直成分をそれぞれｚ倍したものである（ｕＬ(ｙ＋ｗ)，ｖＬ(ｙ＋ｗ)）を、
ｕＬ(ｙ＋ｗ)＝（（（ｑ・ｗｄ－ｙ・ｗｄ－ｗｎ）ｕ0＋（ｙ・ｗｄ＋ｗｎ）ｕ2）ｚ）

////（q・k・wd)

vL(y＋w)＝（（（q・wd－y・wd－wn) v0＋（y・wd＋wn) v2) z）

////（q・k・wd)

を計算することにより（ただし、「////」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等）求め、

さらに座標（i＋p，y＋w）に位置する点の動きベクトルの水平・垂直成分をそれぞれz倍したものである（uR(y＋w)，vR(y＋w)）を、

uR(y＋w)＝（（（q・wd－y・wd－wn) u1＋（y・wd＋wn) u3) z）

////（q・k・wd)

vR(y＋w)＝（（（q・wd－y・wd－wn) v1＋（y・wd＋wn) v3) z）

////（q・k・wd)

を計算することにより求めた後に、

座標（x＋w，y＋w）に位置する画素の動きベクトルの水平・垂直成分をそれぞれm倍したものである（u(x＋w，y＋w)，v(x＋w，y＋w)）を

u(x＋w，y＋w)＝（（（p・wd－x・wd－wn) uL(y＋w)＋（x・wd＋wn) uR(y＋w)) m）//（p・z・wd)

v(x＋w，y＋w)＝（（（p・wd－x・wd－wn) vL(y＋w)＋（x・wd＋wn) vR(y＋w)) m）//（p・z・wd)

を計算することによって（ただし、「//」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等）求めることを特徴とする請求項１に記載のフレーム間予測画像の合成方法。

【請求項４】

座標（i，j）、（i＋p，j）、（i，j＋q）、（i＋p，j＋q）に位置する代表点の動きベクトルの水平・垂直成分をk倍したものである（u0，v0）、（u1，v1）、（u2，v2）、（u3，v3）を用いて座標（x＋w，y＋w）に位置する画素の動きベクトルを求めるときに、

座標（x＋w，j）に位置する点の動きベクトルの水平・垂直成分をそれぞれz

倍したものである（uT(x＋w)，vT(x＋w)）を、

uT(x＋w)＝（（（p・wd－x・wd－wn) u0＋（x・wd＋wn) u1) z）////（p・k・wd)

vT(x＋w)＝（（（p・wd－x・wd－wn) v0＋（x・wd＋wn) v1) z）////（p・k・wd)

を計算することにより（ただし、「////」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等）求め、

さらに座標（x＋w，j＋p）に位置する点の動きベクトルの水平・垂直成分をそれぞれz倍したものである（uB(y＋w)，vB(y＋w)）を、

uB(x＋w)＝（（（p・wd－x・wd－wn) u2＋（x・wd＋wn) u3) z）////（p・k・wd)

vB(x＋w)＝（（（p・wd－x・wd－wn) v2＋（x・wd＋wn) v3) z）////（p・k・wd)

を計算することにより求めた後に、

座標（x＋w，y＋w）に位置する画素の動きベクトルの水平・垂直成分をそれぞれm倍したものである（u(x＋w，y＋w)，v(x＋w，y＋w)）を

u(x＋w，y＋w)＝（（（q・wd－y・wd－wn) uT(x＋w)＋（y・wd＋wn) uB(x＋w)) m）//（q・z・wd)

v(x＋w，y＋w)＝（（（q・wd－y・wd－wn) vT(x＋w)＋（y・wd＋wn) vB(x＋w)) m）//（q・z・wd)

を計算することによって（ただし、「//」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等）求めることを特徴とする請求項２に記載のフレーム間予測画像の合成方法。

【請求項５】

ｐの絶対値が２のα乗（αは負ではない整数）であることを特徴とする請求項１又は３に記載のフレーム間予測画像の合成方法。

【請求項６】

ｑの絶対値が２のβ乗（βは負ではない整数）であることを特徴とする請求項２又は４に記載のフレーム間予測画像の合成方法。

【請求項７】

ｐとｑの絶対値がそれぞれ２のα乗とβ乗（α、βは負ではない整数）であることを特徴とする請求項１又は３に記載のフレーム間予測画像の合成方法。

【請求項８】

ｐとｑの絶対値がそれぞれ２のα乗とβ乗（α、βは負ではない整数）であることを特徴とする請求項２又は４に記載のフレーム間予測画像の合成方法。

【請求項９】

α＋ｈｚが８の正の整数倍であり、かつｗが０であることを特徴とする請求項５又は７に記載のフレーム間予測画像の合成方法。

【請求項１０】

β＋ｈｚが８の正の整数倍であり、かつｗが０であることを特徴とする請求項６又は８に記載のフレーム間予測画像の合成方法。

【請求項１１】

α＋ｈｚ＋ｈｗが８の正の整数倍であり、かつｗ＞０であることを特徴とする請求項５又は７に記載のフレーム間予測画像の合成方法。

【請求項１２】

β＋ｈｚ＋ｈｗが８の正の整数倍であり、かつｗ＞０であることを特徴とする請求項６又は８に記載のフレーム間予測画像の合成方法。

【請求項１３】

複数の異なるαの値に対応し、α＋ｈｚが１６以下となるようにｈｚの値をαの値に応じて変化させることを特徴とする請求項９に記載のフレーム間予測画像の合成方法。

【請求項１４】

複数の異なるβの値に対応し、β＋ｈｚが１６以下となるようにｈｚの値をβの値に応じて変化させることを特徴とする請求項１０に記載のフレーム間予測画像の合成方法。

【請求項１５】

複数の異なるαの値に対応し、α＋ｈｚ＋ｈｗが１６以下となるようにｈｚの値をαの値に応じて変化させることを特徴とする請求項１１に記載のフレーム間予測画像の合成方法。

【請求項１６】

複数の異なるβの値に対応し、β＋ｈｚ＋ｈｗが１６以下となるようにｈｚの値をβの値に応じて変化させることを特徴とする請求項１２に記載のフレーム間予測画像の合成方法。

【請求項１７】

ｚ≧ｍであることを特徴とする請求項１ないし１６に記載のフレーム間予測画像の合成方法。

【請求項１８】

ｋ≧ｚであることを特徴とする請求項１ないし１７に記載のフレーム間予測画像の合成方法。

【請求項１９】

ｐとｑの絶対値がそれぞれ画像の水平と垂直の画素数と異なることを特徴とする請求項１ないし１８に記載のフレーム間予測画像の合成方法。

【請求項２０】

ｒを画像の水平方向の画素数、ｓを画像の垂直方向の画素数として（ただし、ｒとｓは正の整数）、ｐの絶対値を１／２倍した値はｒより小さく、かつｐの絶対値はｒ以上で、かつｑの絶対値を１／２倍した値はｓより小さく、かつｑの絶対値はｓ以上であることを特徴とする請求項１ないし１９に記載のフレーム間予測画像の合成方法。

【請求項２１】

ｒを画像の水平方向の画素数、ｓを画像の垂直方向の画素数とし（ただし、ｒとｓは正の整数）て、ｐの絶対値はｒ以下であり、かつｐの絶対値を２倍した値はｒより大きく、かつｑの絶対値はｓ以下であり、かつｑの絶対値を２倍した値はｓより大きいことを特徴とする請求項１ないし１９に記載のフレーム間予測画像の合成方法。

【請求項２２】

画像の水平方向と垂直方向の画素数がそれぞれｒとｓであり（ただし、ｒとｓは正の整数）、かつ画像の画素が水平座標が０以上ｒ未満、垂直座標が０以上ｓ未満の範囲に存在しているときに、

座標（－ｃ，－ｃ）、（ｒ－ｃ，－ｃ）、（－ｃ，ｓ－ｃ）、（ｒ－ｃ，ｓ－ｃ）に位置する画像の隅の点上に存在し（ただし、ｃ＝ｃｎ／ｃｄ、かつｃｎは負ではない整数、かつｃｄは正の整数、かつｃｎ＜ｃｄ）、水平・垂直成分が１／ｎの整数倍の値をとる動きベクトル（ただし、ｎは正の整数）をｎ倍したものである（ｕ００，ｖ００）、（ｕ０１，ｖ０１）、（ｕ０２，ｖ０２）、（ｕ０３，ｖ０３）（ただし、ｕ００、ｖ００、ｕ０１、ｖ０１、ｕ０２、ｖ０２、ｕ０３、ｖ０３は整数）を用いて、

ｕ’（ｘ，ｙ）＝（（（ｓ・ｃｄ－ｃｎ－ｙ・ｃｄ）（（ｒ・ｃｄ－ｃｎ－ｘ・ｃｄ）ｕ００＋（ｘ・ｃｄ＋ｃｎ）ｕ０１）＋（ｙ・ｃｄ＋ｃｎ）（（ｒ・ｃｄ－ｃｎ－ｘ・ｃｄ）ｕ０２＋（ｘ・ｃｄ＋ｃｎ）ｕ０３））ｋ）／／／（ｒ・ｓ・ｎ・ｃｄ），

ｖ’（ｘ，ｙ）＝（（（ｓ・ｃｄ－ｃｎ－ｙ・ｃｄ）（（ｒ・ｃｄ－ｃｎ－ｘ・ｃｄ）ｖ００＋（ｘ・ｃｄ＋ｃｎ）ｖ０１）＋（ｙ・ｃｄ＋ｃｎ）（（ｒ・ｃｄ－ｃｎ－ｘ・ｃｄ）ｖ０２＋（ｘ・ｃｄ＋ｃｎ）ｖ０３））ｋ）／／／（ｒ・ｓ・ｎ・ｃｄ・ｃｄ），

ｕ０＝ｕ’（ｉ，ｊ），

ｖ０＝ｖ’（ｉ，ｊ），

ｕ１＝ｕ’（ｉ＋ｐ，ｊ），

ｖ１＝ｖ’（ｉ＋ｐ，ｊ），

ｕ２＝ｕ’（ｉ，ｊ＋ｑ），

ｖ２＝ｖ’（ｉ，ｊ＋ｑ），

ｕ３＝ｕ’（ｉ＋ｐ，ｊ＋ｑ），

ｖ３＝ｖ’（ｉ＋ｐ，ｊ＋ｑ），

で表される（ｕ０，ｖ０）、（ｕ１，ｖ１）、（ｕ２，ｖ２）、（ｕ３，ｖ３）を（ただし、「／／／」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等）、代表点（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）の動きベクトルの水平・垂直成分をｋ倍したものとして使用することを特徴とする請求項１ないし

２１に記載のフレーム間予測画像の合成方法。

【請求項２３】

請求項１ないし２２に記載のフレーム間予測画像の合成方法を用いる画像の符号化方法。

【請求項２４】

請求項１ないし２２に記載のフレーム間予測画像の合成方法を用いる画像の復号化方法。

【請求項２５】

代表点の動きベクトルに関する情報を直接符号化することを特徴とする、請求項１ないし２１に記載のフレーム間予測画像の合成方法を用いる画像の符号化方法。

【請求項２６】

符号化データとして直接符号化されている代表点の動きベクトルに関する情報を再生して用いることを特徴とする、請求項１ないし２１に記載のフレーム間予測画像の合成方法を用いる画像の復号化方法。

【請求項２７】

上記画像の隅の点の動きベクトルに関する情報を直接符号化することを特徴とする、請求項２２に記載のフレーム間予測画像の合成方法を用いる画像の符号化方法。

【請求項２８】

符号化データとして直接符号化されている上記画像の隅の点の動きベクトルに関する情報を再生して用いることを特徴とする、請求項２２に記載のフレーム間予測画像の合成方法を用いる画像の復号化方法。

【請求項２９】

請求項２３又は２５又は２７に記載の画像符号化方法を実行する画像符号化装置。

【請求項３０】

請求項２４又は２６又は２８に記載の画像復号化方法を実行する画像復号化装置。

【請求項３１】

請求項１ないし２２に記載のフレーム間予測画像の合成方法を実行するためのソフトウェアを記録した蓄積メディア。

【請求項３２】

請求項２３又は２５又は２７に記載の画像符号化方法を実行するためのソフトウェアを記録した蓄積メディア。

【請求項３３】

請求項２４又は２６又は２８に記載の画像復号化方法を実行するためのソフトウェアを記録した蓄積メディア。

【請求項３４】

請求項２３又は２５又は２７に記載の画像符号化方法によって生成された圧縮ビットストリームを記録した蓄積メディア。

【請求項３５】

請求項２４又は２６又は２８に記載の画像復号化方法によって復号化することができる圧縮ビットストリームを記録した蓄積メディア。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、画像符号化、復号化の処理において、代表点の動きベクトルに対して内・外挿処理を行うことにより画像内の画素の動きベクトルを計算するフレーム間予測画像の合成方法に関するものである。

【０００２】

【従来の技術】

動画像の高能率符号化において、異なる時間に発生したフレーム間の類似性を活用するフレーム間予測（動き補償）は情報圧縮に大きな効果を示すことが知られている。現在の画像符号化技術の主流となっている動き補償方式は、動画像符号化方式の国際標準であるＨ．２６１、ＭＰＥＧ１、ＭＰＥＧ２に採用されているブロックマッチングである。この方式では、符号化しようとする画像を多数のブロックに分割し、ブロックごとにその動きベクトルを求める。

【０００３】

ブロックマッチングは現在最も広く利用されている動き補償方式であるが、画像全体が拡大・縮小・回転している場合には、すべてのブロックに対して動きベクトルを伝送しなければならず、符号化効率が悪くなる問題が発生する。この問題に対し、画像全体の動きベクトル場を少ないパラメータを用いて表現するグローバル動き補償（例えば、M.Hotter, "Differential estimation of the global motion parameters zoom and pan", Signal Processing, vol. 16, no. 3, pp. 249-265, Mar. 1989）が提案されている。これは、画像内の画素（ｘ，ｙ）の動きベクトル（ｕｇ（ｘ，ｙ），ｖｇ（ｘ，ｙ））を、

【０００４】

【数１】

$$u_g(x, y) = a_0x + a_1y + a_2$$
$$v_g(x, y) = a_3x + a_4y + a_5 \qquad \cdots (数1)$$

【０００５】

や、

【０００６】

【数２】

$$u_g(x, y) = b_0xy + b_1x + b_2y + b_3$$
$$v_g(x, y) = b_4xy + b_5x + b_6y + b_7 \qquad \cdots (数2)$$

【０００７】

の形式で表し、この動きベクトルを利用して動き補償を行う方式である。ここでａ0～ａ5、ｂ0～ｂ7は動きパラメータである。動き補償を行う際には、送信側と受信側で同じ予測画像が得られなければならない。このために、送信側は受信側へａ0～ａ5又はｂ0～ｂ7の値を直接伝送しても良いが、代わりにいくつかの代表点の動きベクトルを伝送する方法もある。いま、画像の左上端、右上端、左下端、右下端の画素の座標がそれぞれ（０，０）、（ｒ，０）、（０，ｓ）、（ｒ，ｓ）で表されるとする（ただし、ｒとｓは正の整数）。このとき、代表点（０，０）、（ｒ，０）、（０，ｓ）の動きベクトルの水平・垂直成分をそれぞれ（ｕ

a，va）、（ub，vb）、（uc，vc）とすると、数1は

【0008】

【数3】

$$u_g(x, y) = \frac{u_b - u_a}{r} x + \frac{u_c - u_a}{s} y + u_a$$
$$v_g(x, y) = \frac{v_b - v_a}{r} x + \frac{v_c - v_a}{s} y + v_a \qquad \cdots (数3)$$

【0009】

と書き換えることができる。このことはa0～a5を伝送する代わりにua、va、ub、vb、uc、vcを伝送しても同様の機能が実現できることを意味する。これと同じように、4個の代表点（0，0）、（r，0）、（0，s）、（r，s）の動きベクトルの水平・垂直成分（ua，va）、（ub，vb）、（uc，vc）、（ud，vd）を用いて数2は、

【0010】

【数4】

$$u_g(x, y) = \frac{s-y}{s}\left(\frac{r-x}{r} u_a + \frac{x}{r} u_b\right) + \frac{y}{s}\left(\frac{r-x}{r} u_c + \frac{x}{r} u_d\right)$$
$$= \frac{u_a - u_b - u_c + u_d}{rs} xy + \frac{-u_a + u_b}{r} x + \frac{-u_a + u_c}{s} y + u_a \qquad \cdots (数4)$$
$$v_g(x, y) = \frac{v_a - v_b - v_c + v_d}{rs} xy + \frac{-v_a + v_b}{r} x + \frac{-v_a + v_c}{s} y + v_a$$

【0011】

と書き換えることができる。したがって、b0～b7を伝送する代わりにua、va、ub、vb、uc、vc、ud、vdを伝送しても同様の機能が実現できる。この様子を図1に示す。現フレームの原画像102と参照画像101の間でグローバル動き補償が行われたとして、動きパラメータの代わりに代表点103、104、105、106の動きベクトル107、108、109、110（このとき、動きベクトルは現フレームの原画像の点を出発点として、参照画像内の対応する点を終点とするものとして定義する）を伝送しても良い。本明細書では数1を用いる方式を線形内・外挿に基づくグローバル動き補償、数2を用いる方式を共1次内・外挿に基づくグローバル動き補償とよぶこととする。

【００１２】

このグローバル動き補償の処理を、画像内のより小さい領域に適用するのがワーピング予測である。図２に共一次内・外挿を用いるワーピング予測の例を示す。この図は、参照画像２０１を用いて現フレームの原画像２０２の予測画像を合成する処理を示したものである。まず現フレームは複数の多角形のパッチに分割され、パッチ分割された画像２０９となる。パッチの頂点は格子点とよばれ、各格子点は複数のパッチに共有される。例えば、パッチ２１０は、格子点２１１、２１２、２１３、２１４から構成され、これらの格子点は他のパッチの頂点を兼ねている。このように画像を複数のパッチに分割した後に、動き推定が行なわれる。ここに示す例では、動き推定は各格子点を対象として参照画像との間で行なわれる。この結果、動き推定後の参照画像２０３で各パッチは変形されたものとなる。例えば、パッチ２１０は、変形されたパッチ２０４に対応している。これは、動き推定の結果、格子点２０５、２０６、２０７、２０８がそれぞれ２１１、２１２、２１３、２１４に対応していると推定されたためである。このようにして格子点の動きベクトルを求め、共１次内挿によってパッチ内の各画素の動きベクトルを計算することにより、フレーム間予測画像が合成される。このワーピング予測の処理は基本的に図１に示したグローバル動き補償と同じ処理であり、「画像の隅の点の動きベクトル」が「格子点の動きベクトル」に変えられているだけである。また、長方形の代わりに３角形のパッチを使用すれば、線形内・外挿によるワーピング予測も実現することができる（例えば、特願平06-193970）。

【００１３】

以上述べたグローバル動き補償やワーピング予測を導入することにより、画像の動きを少ないパラメータを用いて正確に表現することが可能となり、より高い情報圧縮率が実現できる。しかし、その一方で符号化および復号化における処理量は従来の方式と比較して増加する。特に数３および４に見られる除算は、処理を複雑にする大きな要因となってしまう。

【００１４】

【発明が解決しようとする課題】

グローバル動き補償やワーピング予測では、予測画像の合成のための処理量が多くなる問題が発生する。本発明の目的は、これらの動き補償方式における除算の処理をビット数の少ないレジスタを用いた２進数のシフト演算に置き換えることにより、演算量を減少させることにある。

【００１５】

【課題を解決するための手段】

グローバル動き補償やワーピング予測を行う際の代表点の座標をうまく選択することによって除算処理をシフト演算で実現できるようにし、さらにシフト演算においてシフトされるビット数を少なくすることにより、ビット数の少ないレジスタによって上記動き補償方式の演算が実現できるようにする。

【００１６】

【発明の実施の形態】

本発明は、グローバル動き補償およびワーピング予測における演算の高速化方法に関する発明（特願平08-060572および特願平08-249601）を応用したものである。また、以下では本発明をグローバル動き補償に適用した場合に関して説明するが、本発明はグローバル動き補償と同様の処理を行うワーピング予測にも応用することが可能である。

【００１７】

以下の議論では、画素のサンプリング間隔が水平・垂直方向共に１であるとして、座標の水平・垂直成分が共に整数にｗを加えた値である点（ただし、ｗ＝ｗｎ／ｗｄ、かつｗｎは負ではない整数、かつｗｄは正の整数、かつｗｎ＜ｗｄ）に画素が存在しているとする。ｗはグローバル動き補償における代表点の座標と画素の座標の位相のずれを表しており、典型的な値としては0、1/2、1/4などが挙げられる。また、画像の水平方向と垂直方向の画素数はそれぞれｒとｓであり（ただし、ｒとｓは正の整数）、かつ画像の画素は水平座標が０以上ｒ未満、垂直座標が０以上ｓ未満の範囲に存在しているとする。

【００１８】

線形内・外挿（アフィン変換）又は共１次内・外挿（共１次変換）を用いた動き補償を行う際には、画素ごとの動きベクトルに対して量子化を行うと、ミスマッチの防止や演算の簡略化などの効果を得ることができる（特願平06-193970）。以下では、画素の動きベクトルの水平成分と垂直成分が１／ｍ（ｍは正の整数）の整数倍であるとする。また、「従来の技術」で説明した代表点の動きベクトルを用いるグローバル動き補償を行うと仮定し、各代表点の動きベクトルは１／ｋ（ｋは正の整数）の整数倍であるとする。なお、本明細書では、「画素の動きベクトル」はグローバル動き補償を行う際に、実際に予測画像を合成するために用いる動きベクトルのことを指す。一方、「代表点の動きベクトル」は画素の動きベクトルを計算するために用いるパラメータを意味している。したがって、量子化ステップサイズの違いなどが原因で、同じ座標上に存在していても画素の動きベクトルと代表点の動きベクトルが一致しない場合も起こり得る。

【００１９】

まず共１次内・外挿を用いるグローバル動き補償について図３を用いて説明する。この例では、図１に示した例のように代表点を画像３０１の隅に位置する点とはせず、（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）に位置する点３０２、３０３、３０４、３０５として一般化している（ｉ、ｊ、ｐ、ｑは整数）。このとき、点３０２、３０３、３０４、３０５は画像の内部に存在していても外部に存在していても良い。代表点の動きベクトルの水平・垂直成分をｋ倍したものをそれぞれ（ｕ0，ｖ0）、（ｕ1，ｖ1）、（ｕ2，ｖ2）、（ｕ3，ｖ3）とすると（ｕ0、ｖ0、ｕ1、ｖ1、ｕ2、ｖ2、ｕ3、ｖ3は整数）、（ｘ＋ｗ、ｙ＋ｗ）に位置する画素の動きベクトルの水平・垂直成分をｍ倍したもの（ｕ（ｘ＋ｗ，ｙ＋ｗ），ｖ（ｘ＋ｗ，ｙ＋ｗ））は、ｗ＝０のときは以下の式で表すことができる（ただし、ｘ、ｙ、ｕ（ｘ，ｙ）、ｖ（ｘ，ｙ）は整数）。

【００２０】

【数５】

$$
\begin{aligned}
u(x+w, y+w) &= u(x, y) \\
&= (((j+q-y)((i+p-x)u_0 + (x-i)u_1) \\
&\quad + (y-j)((i+p-x)u_2 + (x-i)u_3))m) // (pqk) \qquad \cdots (数5) \\
v(x+w, y+w) &= v(x, y) \\
&= (((j+q-y)((i+p-x)v_0 + (x-i)v_1) \\
&\quad + (y-j)((i+p-x)v_2 + (x-i)v_3))m) // (pqk)
\end{aligned}
$$

【００２１】

ただし、「//」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等である。演算誤差を小さくするためには、非整数値は最も近い整数に丸め込まれることが望ましい。このとき整数に１／２を加えた値の丸め込み方法は、

(1) ０に近づける方向に丸め込む、

(2) ０から遠ざける方向に丸め込む、

(3) 被除数が負の場合は０に近づける方向、正の場合は０から遠ざける方向に丸め込む（除数は常に正であるとする）、

(4) 被除数が負の場合は０から遠ざける方向、正の場合は０に近づける方向に丸め込む（除数は常に正であるとする）、

などが考えられる。これらの中で(3)と(4)は、被除数の正負に関わらず丸め込みの方向が変化しないため、正負判定が必要ない分だけ処理量の点で有利である。(3)を用いた高速処理は、例えば以下の式によって実現することができる。

【００２２】

【数６】

$$u(x+w, y+w) = u(x, y)$$
$$= (Lpqk + ((j+q-y)((i+p-x)u_0 + (x-i)u_1) + (y-j)((i+p-x)u_2 + (x-i)u_3))m + ((pqk)\#2))\#(pqk) - L \quad \cdots (数６)$$
$$v(x+w, y+w) = v(x, y)$$
$$= (Mpqk + ((j+q-y)((i+p-x)v_0 + (x-i)v_1) + (y-j)((i+p-x)v_2 + (x-i)v_3))m + ((pqk)\#2))\#(pqk) - M$$

【００２３】

ただし、「#」は小数点以下を０の方向に切り捨てる整数の除算であり、演算の優先順位は乗除算と同じであるとする。これは、一般に計算機では最も実現しやすい形式の除算である。ここで、ＬとＭは除算の被除数を常に正に保つための数で、十分に大きな正の整数である。また、（ｐｑｋ＃２）の項は、除算結果を最も近い整数に丸め込むために用いられる。

【００２４】

処理を整数化することはそれ自体処理量の低減に貢献するが、ここでｐ、ｑ、ｋをそれぞれ２のα、β、ｈｋ乗（α、β、ｈｋは負ではない整数）とすると、数５の除算はα＋β＋ｈｋビットのシフト演算で実現できるため、計算機や専用ハードウェアにおける処理量を大きく減らすことができる。さらにｍを２のｈｍ乗とすれば（ｈｍは負ではない整数、ｈｍ＜α＋β＋ｈｋ）、数６は、

【００２５】

【数７】

$$
\begin{aligned}
u(x+w, y+w) &= u(x, y) \\
&= \Big((2L+1)<<\big(\alpha+\beta+h_k-h_m-1\big) \\
&\quad + (j+q-y)\big((i+p-x)\,u_0 + (x-i)\,u_1\big) \\
&\quad + (y-j)\big((i+p-x)\,u_2 + (x-i)\,u_3\big)\Big) \\
&\quad >> \big(\alpha+\beta+h_k-h_m\big) - L \qquad \cdots (数7) \\
v(x+w, y+w) &= v(x, y) \\
&= \Big((2M+1)<<\big(\alpha+\beta+h_k-h_m-1\big) \\
&\quad + (j+q-y)\big((i+p-x)\,v_0 + (x-i)\,v_1\big) \\
&\quad + (y-j)\big((i+p-x)\,v_2 + (x-i)\,v_3\big)\Big) \\
&\quad >> \big(\alpha+\beta+h_k-h_m\big) - M
\end{aligned}
$$

【００２６】

と書き換えることができ（「x<<α」はxをαビット左にシフトして下位αビットに０を入れる、「x>>α」はxをαビット右にシフトして上位αビットに０を入れることを意味し、これらの演算子の優先順位は加減算と乗除算の中間であるとする）、シフトされるビット数をα＋β＋hk－hmとすることができる。

【００２７】

wが０ではないときには、w＝wn／wdの定義にしたがい、数５は以下のように書き換えることができる。

【００２８】

【数８】

$$u(x+w, y+w) = u\left(x+\frac{w_d}{w_n}, y+\frac{w_d}{w_n}\right)$$
$$= (((w_d j+w_d q-w_d y-w_n)((w_d i+w_d p-w_d x-w_n) u_0 + (w_d x+w_n-w_d i) u_1) + (w_d y+w_n-w_d j)((w_d i+w_d p-w_d x-w_n) u_2 + (w_d x+w_n-w_d i) u_3)) m) // (w_d^2 pqk)$$

・・・（数８）

$$v(x+w, y+w) = v\left(x+\frac{w_d}{w_n}, y+\frac{w_d}{w_n}\right)$$
$$= (((w_d j+w_d q-w_d y-w_n)((w_d i+w_d p-w_d x-w_n) v_0 + (w_d x+w_n-w_d i) v_1) + (w_d y+w_n-w_d j)((w_d i+w_d p-w_d x-w_n) v_2 + (w_d x+w_n-w_d i) v_3)) m) // (w_d^2 pqk)$$

【００２９】

このとき、ｗｄが２のｈｗ乗であり、かつｈｗは負ではない整数であるとすれば、（ｐ・ｑ・ｋ・ｗｄ・ｗｄ）による除算はα＋β＋ｈｋ＋２ｈｗビットのシフト演算となり、ｗ＝０の場合と同様に除算をシフト演算に置換することが可能となる。また、数７の場合と同様に、ｈｍ＜α＋β＋ｈｋ＋２ｈｗであれば、分母、分子の両方をｍで割ることによってシフトされるビット数をα＋β＋ｈｋ＋２ｈｗ－ｈｍビットに減らすことが可能となる。このように、ｗｄが２のｈｗ乗でありさえすれば、ｗ＝０の場合の処理とｗ≠０の場合の処理は本質的に同じである。以下本明細書では、数式が多少複雑となるが、ｗ≠０の場合について検討を行う。ｗ＝０の場合の計算結果を求めるためには、ｗｎ＝０、ｗｄ＝１、ｈｗ＝０を代入すれば良い。

【００３０】

送信側と受信側で同じグローバル動き補償予測画像を得るためには、代表点の動きベクトルに関する情報を何らかの形で受信側に伝える必要がある。代表点の動きベクトルそのまま伝送する方法もあるが、画像の隅の点の動きベクトルを伝送し、この値から代表点の動きベクトルを計算する方法もある。この方法に関し、以下に説明する。

【００３１】

画像の隅の４個の点（－ｃ，－ｃ）、（ｒ－ｃ，－ｃ）、（－ｃ，ｓ－ｃ）、（ｒ－ｃ，ｓ－ｃ）の動きベクトルが１／ｎ整数倍の値のみとれるとして（ｎは正の整数、ｃ＝ｃｎ／ｃｄ、かつｃｎは負ではない整数、かつｃｄは正の整数、かつｃｎ＜ｃｄ）、これらの水平・垂直成分をｎ倍した（ｕ００，ｖ００）、（ｕ０１，ｖ０１）、（ｕ０２，ｖ０２）、（ｕ０３，ｖ０３）がグローバル動きパラメータとして伝送されるとする。ｃは画像の隅の点と代表点の間の位相のずれを表している。このｃの典型的な値としては0、1/2、1/4などが挙げられる。このとき、点（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）それぞれの動きベクトルの水平・垂直成分をｋ倍したものである（ｕ０，ｖ０）、（ｕ１，ｖ１）、（ｕ２，ｖ２）、（ｕ３，ｖ３）を、

【００３２】

【数９】

$$
\begin{aligned}
u_0 &= u'(i, j) \\
v_0 &= v'(i, j) \\
u_1 &= u'(i+p, j) \\
v_1 &= v'(i+p, j) \\
u_2 &= u'(i, j+q) \\
v_2 &= v'(i, j+q) \\
u_3 &= u'(i+p, j+q) \\
v_3 &= v'(i+p, j+q)
\end{aligned}
\qquad \cdots (数９)
$$

【００３３】

と定義する。ただし、ｕ’（ｘ，ｙ）、ｖ’（ｘ，ｙ）は、数５を変形して、

【００３４】

【数１０】

$$u'(x,y) = (((c_d s - c_n - c_d y)((c_d r - c_n - c_d x)u_{00} + (c_d x + c_n)u_{01}) + (c_d y + c_n)((c_d r - c_n - c_d x)u_{02} + (c_d x + c_n)u_{03}))k) /// (c_d^2 rsn) \quad \cdots (数１０)$$

$$v'(x,y) = (((c_d s - c_n - c_d y)((c_d r - c_n - c_d x)v_{00} + (c_d x + c_n)v_{01}) + (c_d y + c_n)((c_d r - c_n - c_d x)v_{02} + (c_d x + c_n)v_{03}))k) /// (c_d^2 rsn)$$

【００３５】

と定義する。このとき、「///」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等である。こうして（ｕ0，ｖ0）、（ｕ1，ｖ1）、（ｕ2，ｖ2）、（ｕ3，ｖ3）を計算し、（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）を代表点とするグローバル動き補償を行えば、（－ｃ，－ｃ）、（ｒ－ｃ，－ｃ）、（－ｃ，ｓ－ｃ）、（ｒ－ｃ，ｓ－ｃ）を代表点とするグローバル動き補償を近似することができる。このときに、上で述べたようにｐとｑを２の負ではない整数乗とすれば、処理を簡略化することが可能となる。一般的に、数５に示したような計算によって画像内の画素の動きベクトルを求めるときには、外挿の処理を行わないようにすることが望ましい。これは、外挿処理によって代表点の動きベクトルの量子化誤差を増幅しないようにするためである。以上の理由から、代表点は画像内の画素をすべて囲むような形に配置することが望ましい。したがって、ｉ＝ｊ＝ｃ＝０の場合などはｐとｑはｒとｓとほぼ同じか、やや大きめの値をとするのが適当である。しかし、ｐとｑの値をあまり大きくし過ぎると演算に必要なビット数が増加してしまうので注意が必要である。

【００３６】

数９、１０の処理において演算誤差を小さくするためには、「///」は非整数値を最も近い整数に丸め込むことが望ましい。このとき整数に１／２を加えた値の丸め込み方法としては、上で述べた(1)～(4)の方法が考えられる。ただし、数５（画素ごとに計算）の場合と比較して、数１４（１枚の画像で４回のみ計算）

は演算が実行される回数が少ないため、(1)又は(2)の方法を選んだとしても全体の演算量に大きな影響は与えない。

【００３７】

ここまでで示した例のように、ｐとｑの値が２の負ではない整数乗となるようにすれば、グローバル動き補償におけるフレーム間予測画像の合成処理は大幅に簡略化することができる。しかし、ここでもう１つの問題が発生する。例えば、画像符号化における典型的なパラメータとしてｐ＝５１２、ｑ＝５１２、ｋ＝３２、ｍ＝１６、ｗｄ＝２、ｗｎ＝１（ｗ＝0.5）である場合を考えると、$\alpha+\beta+hk+2hw-hm=21$となる。このことは、ｕ（ｘ＋ｗ，ｙ＋ｗ）が２進数で１２ビット以上を必要とする値である場合には、数８の演算を高速に実行するために３３ビット以上のレジスタが必要になることを意味している。ｍ＝１６である場合などには、ｕ（ｘ＋ｗ，ｙ＋ｗ）は実際の動きベクトルの水平成分に１６を掛けた値となるため、これが２進数で１２ビット以上必要な値となるケースは十分にあり得る。しかし、その一方で３３ビット以上の整数を格納できるレジスタを持つプロセッサは現時点では少なく、かつ将来的にも高価となることが予想される。また、一般的にプロセッサの回路規模が大きくなれば、その分だけ消費電力も多くなるため、大きなレジスタを要求するアルゴリズムは消費電力の観点からも不利となる。したがって、除算をシフト演算に置換できた場合でも、シフトされるビット数はできるだけ少ないことが望ましい。

【００３８】

この問題を解決するためには、以下に説明する２段階の処理によるアルゴリズムが有効となる。（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）に位置する代表点の動きベクトルを用いて（ｘ＋ｗ、ｙ＋ｗ）に位置する画素の動きベクトルを計算する前に、まず（ｉ，ｙ＋ｗ）と（ｉ＋ｐ，ｙ＋ｗ）に存在する仮代表点の動きベクトルを、水平・垂直成分が１／ｚの整数倍（ｚは正の整数）となるように求める。上の例と同様に代表点（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）、（ｉ＋ｐ，ｊ＋ｑ）の動きベクトルの水平・垂直成分をｋ倍したものをそれぞれ（ｕ0，ｖ0）、（ｕ1，ｖ1）、（ｕ2，ｖ2）、（ｕ3，ｖ3）とする（ｕ0、ｖ0、ｕ1、ｖ1、ｕ2、ｖ2、ｕ3、ｖ3は整数）。このとき、

（ｉ，ｙ＋ｗ）と（ｉ＋ｐ，ｙ＋ｗ）に仮代表点を配置し、これらの仮代表点の動きベクトルの水平・垂直成分をｚ倍したものである（ｕＬ(ｙ＋ｗ)，ｖＬ(ｙ＋ｗ)）と（ｕＲ(ｙ＋ｗ)，ｖＲ(ｙ＋ｗ)）を、以下のように定義する。

【００３９】

【数１１】

$$
\begin{aligned}
&u_L(y+w) \\
&\quad = (((w_dj+w_dq-w_dy-w_n)\,u_0 + (w_dy+w_n-w_dj)\,u_2)\,z) \,////\, (w_dqk) \\
&v_L(y+w) \\
&\quad = (((w_dj+w_dq-w_dy-w_n)\,v_0 + (w_dy+w_n-w_dj)\,v_2)\,z) \,////\, (w_dqk) \\
&u_R(y+w) \\
&\quad = (((w_dj+w_dq-w_dy-w_n)\,u_1 + (w_dy+w_n-w_dj)\,u_3)\,z) \,////\, (w_dqk) \\
&v_R(y+w) \\
&\quad = (((w_dj+w_dq-w_dy-w_n)\,v_1 + (w_dy+w_n-w_dj)\,v_3)\,z) \,////\, (w_dqk)
\end{aligned}
\qquad \cdots （数１１）
$$

【００４０】

このとき、「////」は通常の除算による演算結果が整数ではない場合にこれを近隣の整数に丸め込む除算で、演算子としての優先順位は乗除算と同等である（この「////」には、上で説明した「///」と同様の機能が要求される）。（ｉ，ｙ＋ｗ）は（ｉ，ｊ）と（ｉ，ｊ＋ｑ）を結んだ線上に存在しているため、（ｕＬ(ｙ＋ｗ)，ｖＬ(ｙ＋ｗ)）は、（ｕ0，ｖ0）と（ｕ2，ｖ2）を用いた１次元の線形内・外挿で容易に求めることができる。また、同様に（ｉ＋ｐ，ｙ＋ｗ）は（ｉ＋ｐ，ｊ）と（ｉ＋ｐ，ｊ＋ｑ）を結んだ線上に存在しているため、同じように１次元の線形内・外挿で求めることができる。

【００４１】

このようにして求めた仮代表点の動きベクトル（ｕＬ(ｙ＋ｗ)，ｖＬ(ｙ＋ｗ)）と（ｕＲ(ｙ＋ｗ)，ｖＲ(ｙ＋ｗ)）に対して１次元の線形内・外挿を行うことにより、（ｘ＋ｗ、ｙ＋ｗ）に存在する画素の動きベクトルの水平・垂直成分をｍ倍したものである（ｕ（ｘ＋ｗ、ｙ＋ｗ），ｖ（ｘ＋ｗ、ｙ＋ｗ））を求める。この処理は、以下の式に従って行われる。

【００４２】

【数１２】

$$u(x+w,y+w) = (((w_d i+w_d p-w_d x-w_n)\, u_L(y+w) + (w_d x+w_n-w_d i)\, u_R(y+w))m) // (w_d pz)$$
$$v(x+w,y+w) = (((w_d i+w_d p-w_d x-w_n)\, v_L(y+w) + (w_d x+w_n-w_d i)\, v_R(y+w))m) // (w_d pz) \quad \cdots (数１２)$$

【００４３】

ここでも上と同様にｐを２のα乗、ｍを２のｈｍ乗、ｚを２のｈｚ乗、ｗｄを２のｈｗ乗（α、ｈｍ、ｈｚ、ｗｄは負ではない整数）とすれば、数１２におけるｐ・ｚ・ｗｄによる除算は、α＋ｈｚ＋ｈｗ－ｈｍビットの右シフト（ただし、ｈｍ＜α＋ｈｚ＋ｈｗの場合）に置換することができる。しかも、ｚ＝１６（ｈｚ＝４）とした上で、上で述べた典型的なパラメータｐ＝５１２、ｑ＝５１２、ｋ＝３２、ｍ＝１６、ｗｄ＝２、ｗｎ＝１（ｗ＝0.5）を使用した場合、シフトされるビット数は１０ビットとなり、演算に用いるレジスタに必要なビット数を大幅に抑えることが可能となる。なお、上の例では、まず代表点の動きベクトルに対して垂直方向の１次元線形内・外挿を行って仮代表点の動きベクトルを求め、この仮代表点の動きベクトルに対して水平方向の１次元線形内・外挿を行って画素の動きベクトルを求めている。これとは逆に、仮代表点の動きベクトルを求める際には水平方向、画素の動きベクトルを求める際には垂直方向の１次元線形内・外挿を行っても同様の機能を実現することができる。

【００４４】

この方式では、画素の動きベクトルを求める際に数１１と数１２の２段階の処理が必要となるため、一見演算量が多くなるように思われる。しかし、一旦仮代表点の動きベクトルを求めてしまえば、これが垂直座標ｙ＋ｗに存在しているライン上のｒ個の画素すべてに対して使用できるため、全体の処理量の中に占める数１１の処理量はきわめて少なくなる。したがって、シフトされるビット数の削減によって得られる利益（＝より小さいレジスタの活用）の影響の方が、数１１の計算を実行する分の演算量の増加による悪影響より大きくなる。

【００４５】

上記処理により（ｕ（ｘ＋ｗ、ｙ＋ｗ），ｖ（ｘ＋ｗ、ｙ＋ｗ））の値が得られた後には、以下の処理によって（ｕ（ｘ＋ｗ、ｙ＋ｗ），ｖ（ｘ＋ｗ、ｙ＋ｗ））を整数部（ｕＩ（ｘ＋ｗ、ｙ＋ｗ），ｖＩ（ｘ＋ｗ、ｙ＋ｗ））と小数部（ｕＦ（ｘ＋ｗ、ｙ＋ｗ），ｖＦ（ｘ＋ｗ、ｙ＋ｗ））に分けることができる。

【００４６】

【数１３】

$$u_I(x+w,y+w) = \{(Lm + u(x+w,y+w)) >> h_m\} - L$$
$$v_I(x+w,y+w) = \{(Mm + v(x+w,y+w)) >> h_m\} - M \qquad \cdots (数１３)$$

【００４７】

【数１４】

$$u_F(x+w,y+w) = u(x+w,y+w) - u_I(x+w,y+w)\,m$$
$$v_F(x+w,y+w) = v(x+w,y+w) - v_I(x+w,y+w)\,m \qquad \cdots (数１４)$$

【００４８】

ただし、ｕＩ（ｘ＋ｗ、ｙ＋ｗ）とｖＩ（ｘ＋ｗ、ｙ＋ｗ）は整数であり、画素の動きベクトルの整数部を表している。一方、ｕＦ（ｘ＋ｗ、ｙ＋ｗ）とｖＦ（ｘ＋ｗ、ｙ＋ｗ)はそれぞれ０以上ｍ未満の値を持つ整数であり、画素の動きベクトルの小数部をｍ倍したものである。なお、上の例と同様にｍは２のｈｍ乗であり（ｈｍは負ではない整数）、ＬとＭはシフトされる値を負の値ではなくするための十分に大きな整数である。

【００４９】

輝度値の内挿方式として共１次内挿が用いられる場合には、さらに以下の処理によってフレーム間予測画像内の画素の輝度値が求められる。ｘ’＝ｘ＋ｗ＋ｕＩ（ｘ＋ｗ、ｙ＋ｗ）、ｙ’＝ｙ＋ｗ＋ｖＩ（ｘ＋ｗ、ｙ＋ｗ）として、参照画像の（ｘ’，ｙ’）、（ｘ’＋１，ｙ’）、（ｘ’，ｙ’＋１）、（ｘ’＋１，ｙ’＋１）に位置する画素の輝度値をそれぞれＹａ、Ｙｂ、Ｙｃ、Ｙｄとすれば、フレ

ーム間予測画像において（ｘ＋ｗ，ｙ＋ｗ）に位置する画素の輝度値Ｙ（ｘ＋ｗ、ｙ＋ｗ）は、

【００５０】

【数１５】

$$Y(x+w,y+w) = \{(m-v_F)((m-u_F)Y_a + u_F Y_b) + v_F((m-u_F)Y_c + u_F Y_d) + (m^2>>1)\} >> (2h_m) \quad \cdots (数15)$$

【００５１】

によって求められる。ただし、ｕＦ、ｖＦはそれぞれｕＦ(ｘ＋ｗ，ｙ＋ｗ)、ｖＦ(ｘ＋ｗ，ｙ＋ｗ)の略号である。

【００５２】

数１２と数１３では、それぞれα＋ｈｚ＋ｈｗ－ｈｍビットとｈｍビットの右シフトが行われる。このことは、数１０の計算の際に（α＋ｈｚ＋ｈｗ－ｈｍ）＋ｈｍ＝α＋ｈｚ＋ｈｗビットのシフトを行ってしまえば一気にｕＩ（ｘ＋ｗ、ｙ＋ｗ）とｖＩ（ｘ＋ｗ、ｙ＋ｗ）を求めることができることを意味している。このとき、α＋ｈｚ＋ｈｗを８の整数倍とすると、実装上便利である。一般的にプロセッサのレジスタは８ビット単位の大きさを持っており、８ビットのレジスタを２個（上位ビットのレジスタと下位ビットのレジスタ）つなげて１６ビットのレジスタとして使用したり、８ビットのレジスタを４個、又は１６ビットのレジスタを２個つなげて３２ビットのレジスタとして使用することができるようになっている場合が多い。ここで例えば１６ビットのシフト演算によってｕＩ（ｘ＋ｗ、ｙ＋ｗ）とｖＩ（ｘ＋ｗ、ｙ＋ｗ）の値が計算されるのであれば、わざわざシフト演算を行う必要はなくなる。つまり、シフトされる前の値を３２ビットのレジスタに格納しておき、その上位１６ビットを独立したレジスタとして使用すれば、その１６ビットレジスタにｕＩ（ｘ＋ｗ、ｙ＋ｗ）又はｖＩ（ｘ＋ｗ、ｙ＋ｗ）の値が格納されていることになる。

【００５３】

もちろん、シフトされるビット数を８の整数倍とすることは、数１０の処理だけでなく本明細書でこれまで述べてきたあらゆるシフト演算に対し、実装を容易

にする効果を持つ。しかし、特に実行される回数の多いシフト演算（例えば画素ごとに実行されるシフト演算）に対して実装を容易にすることは重要である。また、シフトされるビット数が８の整数倍ではない場合でも、分母と分子に事前に同じビット数だけの左シフトを加えておくことにより、除算による右シフトを増やすことは可能である。例えば、６ビットの右シフトによって実現される演算があった場合に、シフトされる数値にあらかじめ４を掛けておく（これは２ビットの左シフトを行ったことに相当する）ことにより、同じ演算を８ビットの右シフトとして実現することが可能となる（数５のｕ（ｘ＋ｗ，ｙ＋ｗ）に関する式を例にとれば、あらかじめｕ0、ｕ1、ｕ2、ｕ3を４倍しておくことにより、この処理を実現することが可能となる）。ただし、このような処理を行う際には、シフトされる数に関してオーバーフローが発生しないように注意する必要がある。

【００５４】

画像符号化装置および画像復号化装置には、複数の画像サイズに対応できるようになっているものが多い。この場合、例えば数１２、１３、１４を用いたグローバル動き補償を実行したときには、画像サイズの変化に応じてシフトされるビット数が変化する現象が起こり、シフトされるビット数を８の整数倍に固定しておくことができなくなる。このような場合、次に述べるような対処法がある。例えば、上の例のようにｕｌ（ｘ＋ｗ、ｙ＋ｗ）とｖｌ（ｘ＋ｗ、ｙ＋ｗ）を求めるためにα＋ｈｚ＋ｈｗビットの右シフトが必要であり、αが７～１１の値をとり得る場合を考える。このとき、

αが１０より小さいときはｈｚ＝５、ｈｗ＝１、

α＝１１のときはｈｚ＝４、ｈｗ＝１

とすれば、シフトされるビット数を常に１６以下とすることができる。上で述べたように、シフトされるビット数が１６より小さい場合には、あらかじめシフトされる数に定数を掛けておくことにより、シフトされるビット数を擬似的に１６ビットとすることが可能である。このように、画像サイズが変化したときに他のパラメータ（例えば動きベクトルの量子化ステップサイズ）もこれに合わせて変化させることにより、シフトされるビット数が都合の良い値となるように制御することができる。しかし、上記の方法を使う場合には、復号画像の画質に著しい

劣化を生じさせるほど、動きベクトルの量子化ステップサイズを大きくしてしまわないように注意する必要がある。

【００５５】

一般的なグローバル動き補償に本明細書で示したアルゴリズムを適用した場合には、まず１／ｎ画素精度の画像の隅の点の動きベクトルを用いて代表点の動きベクトルを１／ｋ画素精度で求め、続いて代表点の動きベクトルを用いて仮代表点の動きベクトルを１／ｚ画素精度で求めた後に、この仮代表点の動きベクトルを用いて画素の動きベクトルが１／ｍ画素精度で求められる。画像の隅の点の動きベクトルが動きパラメータとして伝送される場合には、このパラメータによる共１次内・外挿を正確に近似するという意味で、ｋをできるだけ大きな値にすることが望ましい。しかし、いずれにせよ代表点の動きベクトルの水平・垂直成分には、量子化の影響で１／（２ｋ）以下の絶対値をもつ誤差が含まれることになる。近似を正確にするという意味からは、仮代表点の動きベクトルも、なるべく精度を高くすることが望ましい。しかし、仮代表点の動きベクトルは代表点の動きベクトルを用いて求められるため、代表点の動きベクトル以上の精度を持たせて計算してもあまり意味がない。したがって、演算に必要なビット数を抑える意味でｚ≦ｋとすることが望ましい。また、同様の理由により、ｍ≦ｚとすることが望ましい。

【００５６】

これまで共一次内・外挿を用いたグローバル動き補償に関して説明してきたが、線形内・外挿を用いた場合も同様の処理を導入することによって、シフトされるビット数を制御することができる。例えば、（ｉ，ｊ）、（ｉ＋ｐ，ｊ）、（ｉ，ｊ＋ｑ）に存在する（ｉ、ｊ、ｐ、ｑは整数）代表点の動きベクトルの水平・垂直成分をｋ倍したものをそれぞれ（ｕ0，ｖ0）、（ｕ1，ｖ1）、（ｕ2，ｖ2）とする（ｕ0、ｖ0、ｕ1、ｖ1、ｕ2、ｖ2は整数）。このとき、画素（ｘ＋ｗ，ｙ＋ｗ）の動きベクトルの水平・垂直成分をm倍したもの（ｕ(ｘ＋ｗ，ｙ＋ｗ)，ｖ(ｘ＋ｗ，ｙ＋ｗ)）は以下の式で表すことができる（ただし、ｘ、ｙ、ｕ(ｘ＋ｗ，ｙ＋ｗ)、ｖ(ｘ＋ｗ，ｙ＋ｗ)は整数、ｗの定義は上と同じ）。

【００５７】

【数１６】

$$u(x+w,y+w) = (((u_1-u_0)(w_dx+w_n-w_di)\,q + (u_2-u_0)(w_dy+w_n-w_dj)\,p + u_0w_dpq)\,m) // (w_dpqk)$$
$$v(x+w,y+w) = (((v_1-v_0)(w_dx+w_n-w_di)\,q + (v_2-v_0)(w_dy+w_n-w_dj)\,p + v_0w_dpq)\,m) // (w_dpqk)$$
　　・・・（数１６）

【００５８】

この場合もｐ、ｑ、ｋ、ｍ、ｗｄがそれぞれ２のα乗、β乗、ｈｋ乗、ｈｍ乗、ｈｗ乗であり（α、β、ｈｋ、ｈｍ、ｈｗは負ではない整数）、さらにα≧βであるとすれば、この式は

【００５９】

【数１７】

$$u(x+w,y+w) = (((u_1-u_0)(w_dx+w_n-w_di)\,2^{\alpha-\beta} + (u_2-u_0)(w_dy+w_n-w_dj) + u_0w_dp)\,m) // (w_dpk)$$
$$v(x+w,y+w) = (((v_1-v_0)(w_dx+w_n-w_di)\,2^{\alpha-\beta} + (v_2-v_0)(w_dy+w_n-w_dj) + v_0w_dp)\,m) // (w_dpk)$$
　　・・・（数１７）

【００６０】

と書き換えることができ、共１次内・外挿を用いた場合と同様に、α＋ｈｋ＋ｈｗビットの右シフトにより（ｘ＋ｗ，ｙ＋ｗ）に存在する画素の動きベクトルの整数部を求めることができる。したがって、α＋ｈｋ＋ｈｗが８の整数倍となるようにすれば、上と同様の理由により実装を行いやすくすることができる。なお、α＜βの場合には、シフトされるビット数はβ＋ｈｋ＋ｈｗビットとなる。

【００６１】

本発明は、特願平08-060572に示されている従来型の専用回路・専用チップを用いる画像符号化装置、画像復号化装置の他に、汎用プロセッサを用いるソフトウェア画像符号化装置、ソフトウェア画像復号化装置にも適用することができる。図４と５にこのソフトウェア画像符号化装置４００とソフトウェア画像復号化

装置５００の例を示す。ソフトウェア符号化器４００では、まず入力画像４０１は入力フレームメモリ４０２に蓄えられ、汎用プロセッサ４０３はここから情報を読み込んで符号化の処理を行う。この汎用プロセッサを駆動するためのプログラムはハードディスクやフロッピーディスクなどによる蓄積デバイス４０８から読み出されてプログラム用メモリ４０４に蓄えられる。また、汎用プロセッサは処理用メモリ４０５を活用して符号化の処理を行う。汎用プロセッサが出力する符号化情報は一旦出力バッファ４０６に蓄えられた後に符号化ビットストリーム４０７として出力される。

【００６２】

一方、ソフトウェア復号化器５００では、入力された符号化ビットストリーム５０１は一旦入力バッファ５０２に蓄えられた後に汎用プロセッサ５０３に読み込まれる。汎用プロセッサはハードディスクやフロッピーディスクなどによる蓄積デバイス５０８から読み出されたプログラムを蓄えるプログラム用メモリ５０４、および処理用メモリ５０５を活用して復号化処理を行う。この結果得られた復号化画像は一旦出力フレームメモリ５０６に蓄えられた後に出力画像５０７として出力される。

【００６３】

図４と５に示したソフトウェア画像符号化装置、ソフトウェア画像復号化装置に本明細書で示したフレーム間予測画像の合成方法を実行するプログラムを実行させると、グローバル動き補償やワーピング予測の処理をより少ない演算量で実現することが可能となる。このため、本発明を用いない場合と比較して、消費電力の低減、装置の低価格化、より大きな画像を実時間で処理できるようになる、画像符号化・復号化以外の処理を含む同時並列処理を行うことが可能となる、等の効果を期待することができる。また、本明細書で示したアルゴリズムを用いることにより、従来の画像復号化装置では演算能力の限界から実時間で再生できなかったような圧縮画像データを、実時間で再生することが可能となる。

【００６４】

なお、以下の変形も本発明に含まれることは明らかである。

【0065】

（1）従来型の画像符号化方法では、フレーム間予測を行った後に離散コサイン変換などによる誤差符号化が行われるが、フレーム間予測画像をそのまま再生画像として使用する画像符号化方法・復号化方法に対しても、本発明は有効である。

【0066】

（2）本明細書では、画像の形状は長方形であることを仮定したが、長方形以外の任意の形状を持つ画像にも、本発明は適用可能である。この場合、まず任意形状の画像を囲む長方形に対して本発明の処理を適用し、任意形状画像に含まれる画素に対してのみ動きベクトルを求める演算を行えば良い。

【0067】

（3）本明細書では、ｐ又はｑの値が2の負ではない整数乗であることを前提として2段階の処理による動きベクトルの内・外挿アルゴリズムを示した。しかし、ｐおよびｑが2の負ではない整数乗ではない場合でも、この2段階処理アルゴリズムは除算における分母の値を小さくするという効果を持っており、レジスタのオーバーフローを防ぐ意味で有効である。

【0068】

【発明の効果】

本発明により、グローバル動き補償やワーピング予測における予測画像合成処理を簡略化することが可能となり、ソフトウェアや専用ハードウェアによる処理の負担を軽くすることが可能となる。

【図面の簡単な説明】

【図1】

代表点の動きベクトルを伝送するグローバル動き補償の例を示した図。

【図2】

ワーピング予測の処理例を示した図。

【図3】

高速な処理を行うための代表点の配置の例を示した図。

【図４】

ソフトウェア画像符号化装置の構成例を示した図。

【図５】

ソフトウェア画像復号化装置の構成例を示した図。

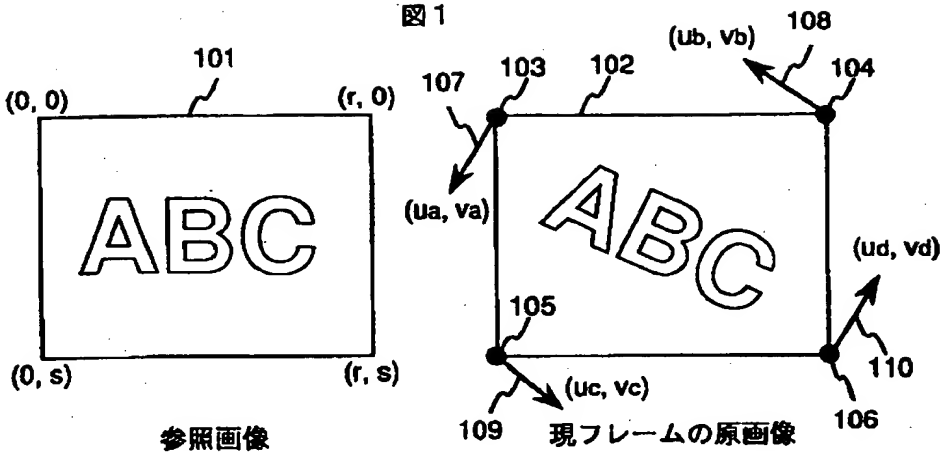
【符号の説明】

１０１、２０１…参照画像、１０２、２０２…現フレームの原画像、１０３～１０６、３０２～３０５…代表点、１０７～１１０…代表点の動きベクトル、２０３…動き推定後の参照画像のパッチと格子点、１０４、２１０…パッチ、２０５～２０８、２１１～２１４…格子点、２０９…現フレームの原画像のパッチと格子点、３０１…フレーム間予測画像、４００…ソフトウェア画像符号化器、４０２…入力フレームメモリ、４０３、５０３…汎用プロセッサ、４０４、５０４…プログラム用メモリ、４０５、５０５…処理用メモリ、４０６…出力バッファ、４０７、５０１…符号化ビットストリーム、４０８、５０８…蓄積デバイス、５００…ソフトウェア画像復号化器、５０２…入力バッファ、５０６…出力フレームメモリ。

【書類名】図面

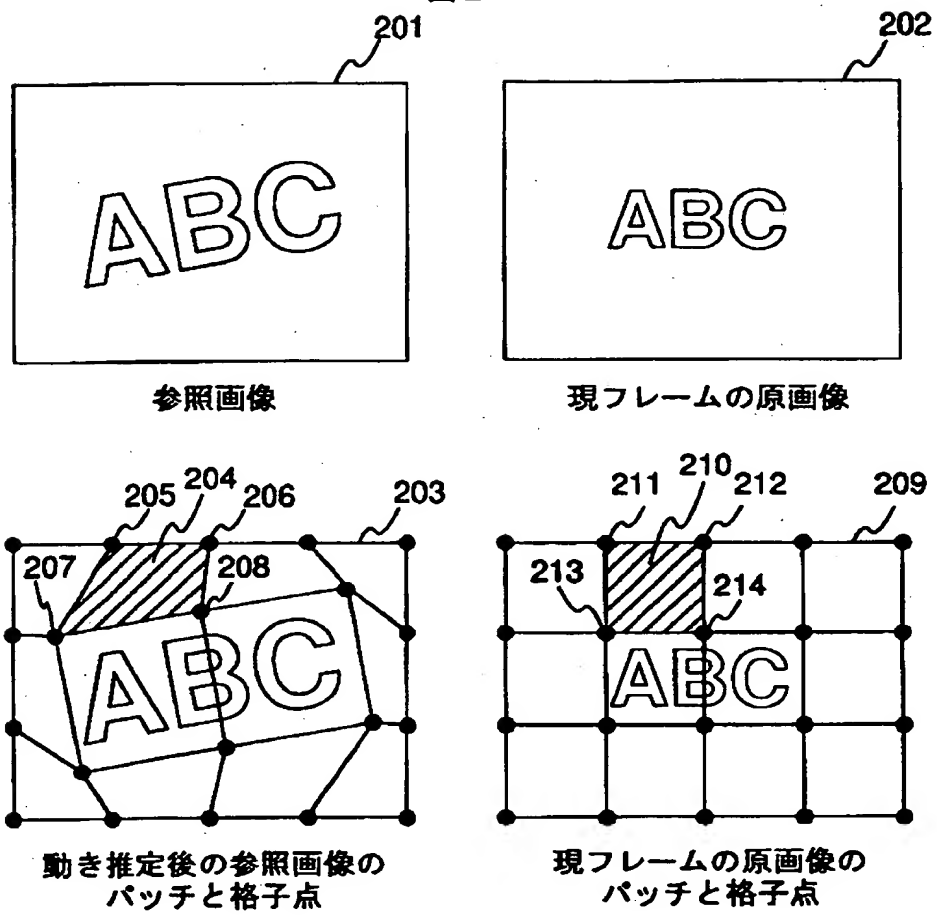
【図１】

図１


101
(0, 0)
(r, 0)
ABC
(0, s)
(r, s)
参照画像
107
103
102
(ub, vb)
108
104
(ua, va)
ABC
(ud, vd)
105
110
(uc, vc)
109
106
現フレームの原画像

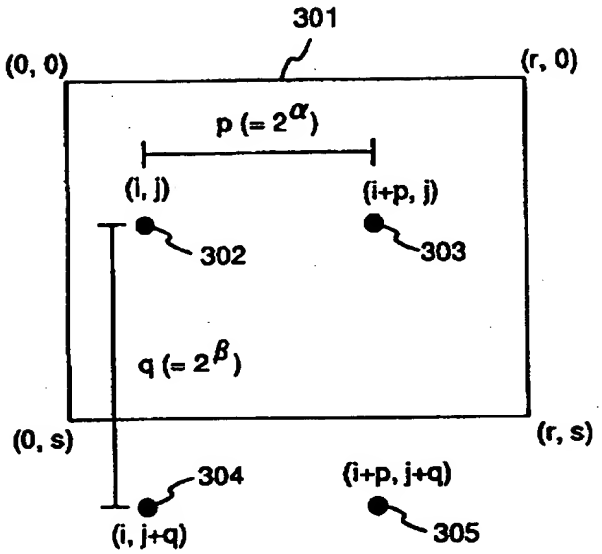
【図２】

図２


201
202
ABC
ABC
参照画像
現フレームの原画像
205
204
206
203
207
208
ABC
211
210
212
209
213
214
ABC
動き推定後の参照画像の
パッチと格子点
現フレームの原画像の
パッチと格子点

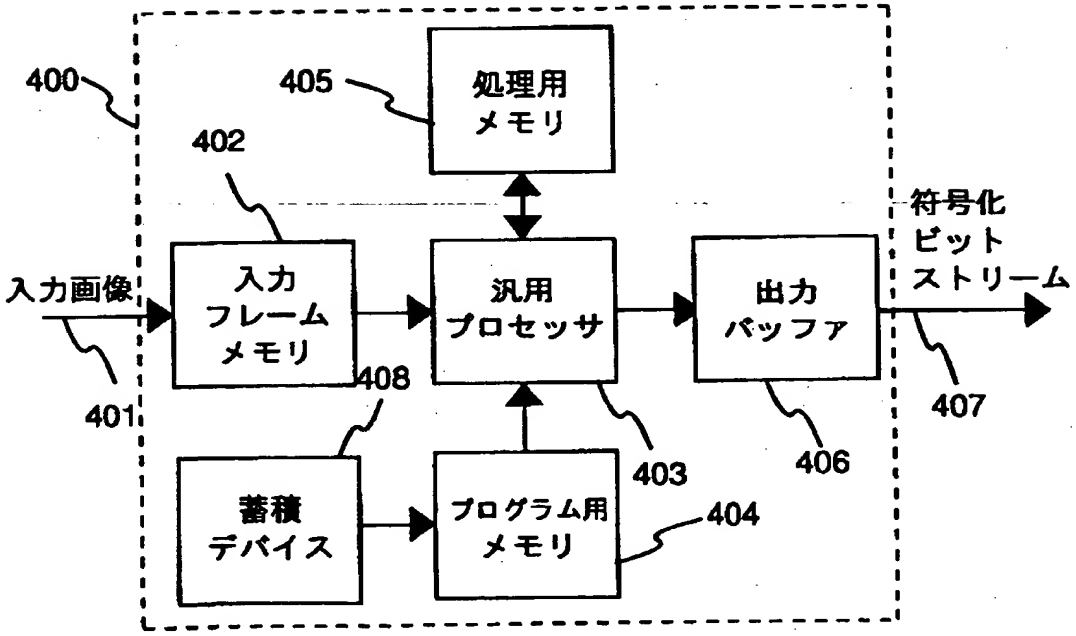
【図３】

図３


301
(0, 0)
(r, 0)
p (= 2^α)
(i, j)
(i+p, j)
302
303
q (= 2^β)
(0, s)
(r, s)
304
(i+p, j+q)
(i, j+q)
305

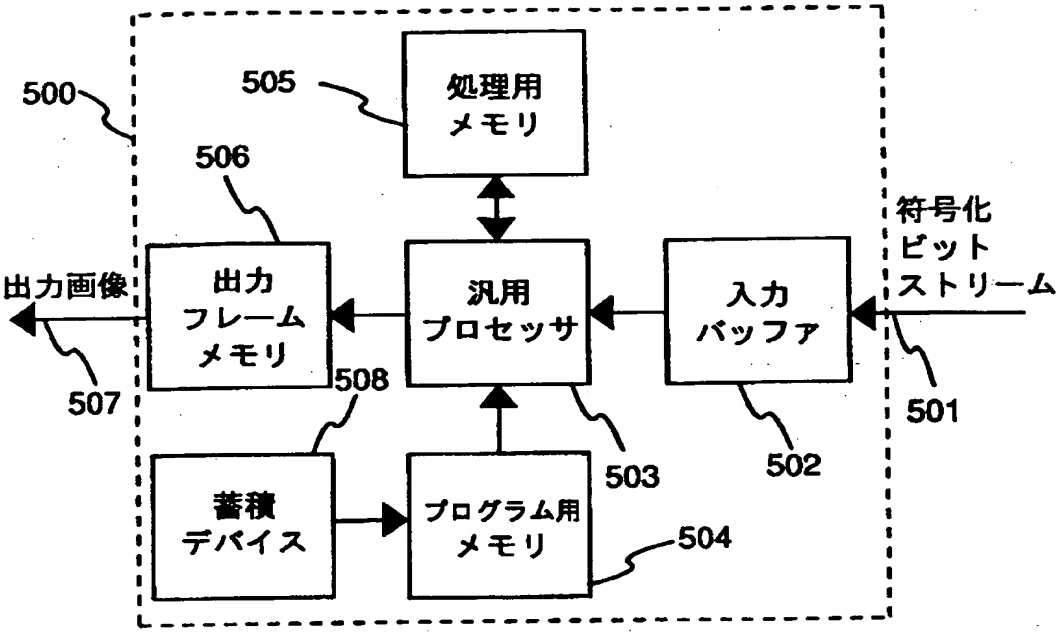
【図４】

図４


400
405
処理用
メモリ
402
符号化
ビット
ストリーム
入力画像
入力
フレーム
メモリ
汎用
プロセッサ
出力
バッファ
401
408
407
403
406
蓄積
デバイス
プログラム用
メモリ
404

【図５】

図５


500
505
処理用
メモリ
506
出力画像
出力
フレーム
メモリ
汎用
プロセッサ
入力
バッファ
符号化
ビット
ストリーム
507
508
503
502
501
蓄積
デバイス
プログラム用
メモリ
504

【書類名】要約書

【要約】

【課題】　グローバル動き補償およびワーピング予測の処理における演算を簡略化する方法を提供する。

【解決手段】　空間的な間隔が特別な特徴をもつ複数の代表点３０２、３０３、３０４、３０５の動きベクトルに対して、２段階の内・外挿処理を行うことによって動きベクトルを求める。

【効果】　予測画像を合成する際の除算がシフトされるビット数が少ないシフト演算によって代用できるため、計算機又は専用ハードウェアによる処理を簡略化することができる。

【選択図】図３

【書類名】　職権訂正データ
【訂正書類】　特許願

＜認定情報・付加情報＞
【特許出願人】
【識別番号】　000005108
【住所又は居所】　東京都千代田区神田駿河台四丁目６番地
【氏名又は名称】　株式会社日立製作所
【代理人】　申請人
【識別番号】　100068504
【住所又は居所】　東京都千代田区丸の内１－５－１　株式会社日立製作所　知的所有権本部内
【氏名又は名称】　小川　勝男

出　願　人　履　歴　情　報

識別番号　　〔０００００５１０８〕

１．変更年月日　　１９９０年　８月３１日
［変更理由］　新規登録
住　所　東京都千代田区神田駿河台４丁目６番地
氏　名　株式会社日立製作所